A Variable Slot Length TDMA Protocol for Personal Communication Systems

Hongjun Zhang (hongjun@cs.ualberta.ca) and Pawel Gburzynski (pawel@cs.ualberta.ca)
University of Alberta, Department of Computing Science

April 7, 2002

Abstract. We present a new TDMA-based scheme intended for carrying traffic with diverse QoS requirements in mobile environments, e.g., Personal Communication Systems (PCS). In contrast to most other TDMA protocols for mobile applications, instead of trying to fit the offered traffic to the slot size, our solution adapts the slot size to the offered traffic. This feature is combined with a dynamic and responsive bandwidth scheduler. As demonstrated by our performance studies, the proposed scheme is more flexible and incurs lower bandwidth overhead than other TDMA-based solutions.

Keywords: TDMA, QoS, wireless ATM

1. Introduction

With the rapid growth of personal networking, the demand for non-voice services in a mobile environment has been growing much more rapidly than those services can be deployed—because of bandwidth limitations and the restrictive nature of traditional, inflexible, voice-oriented solutions. New protocols for personal communication must assume that non-voice traffic is an integral and important part of contemporary mobile networking. They must account for the presence of several different classes of traffic with diverse patterns and quality of service (QoS) requirements, and make sure that those classes coexist as comfortably as possible within the restrictive framework of mobile environments.

In this paper, we introduce a new TDMA-based bandwidth allocation protocol for mobile applications, aimed at accommodating sessions with diverse patterns and priorities. One feature of our protocol is flexible slotting: slot boundaries within the uplink frame are tailored to the transmission requirements of the mobile stations. We demonstrate that this solution, combined with a highly responsive bandwidth scheduler, offers better flexibility of bandwidth allocation than traditional methods based on fixed-size slots, which directly translates into a lower overhead and better fulfillment of the QoS requirements of the diverse traffic types occurring in modern PCS applications.

2. Overview of TDMA-based medium access schemes

Several TDMA-based access protocols (some of them catering to various multimedia services) have been proposed for mobile networks, e.g., PRMA [12], DPRMA [7], C-PRMA [2], DRMA [21], D-TDMA [8, 23], RAMA [1], DQRUMA[16]. All those protocols assume the same model of TDMA whereby a single *uplink* radio channel is shared by all mobile stations trying to transmit packets to the base station. In contrast, the *downlink* channel is used exclusively by the base station to send to the mobile stations packets as well as control information required by the access protocol for the uplink channel. As the base station is the sole agent transmitting on the downlink channel, that channel is never subject to contention.

All the protocols mentioned above use a similar channel structure. Time on the uplink channel is divided into frames, and each frame is divided into a number of equal length transmission slots and a number of possibly smaller minislots used for contention resolution. This approach is efficient if all mobiles have the same traffic patterns and bandwidth requirements, e.g., as in a cellular voice system. In a scenario with diverse applications involving VBR and data traffic, some protocols (e.g., D-TDMA and RAMA) have the flexibility to assign two or more transmission slots to one mobile station, as needed to satisfy its dynamic bandwidth requirements. One problem with this solution is that the granularity of bandwidth assignment is constrained by the slot size or, as in DPRMA, by some multiples of slot size. Since the slot length is usually tailored to efficiently accommodate one traffic type, i.e., voice, it may not fit very well the requirements of non-CBR traffic.

More recent variations on those themes, e.g., [9, 10, 11, 15, 24], follow essentially the same paradigm, focusing on bandwidth scheduling and analytical performance studies. In recognition of the critical role of the bandwidth scheduler, which must announce the layout of the next uplink frame immediately after the end of the previous frame, the protocols discussed in [10, 11, 15] assume that the uplink and downlink frames are time-multiplexed on the same radio channel, with dynamically adjustable boundaries. All those solutions assume that bandwidth on the uplink channel is assigned in fixed-size slots, usually corresponding to ATM cells. A good review of seven TDMA-based protocols is given in [17].

Even the flexible schemes that can allocate multiple slots to a single source suffer from a bandwidth wastage resulting from the fact that each of the multiple slots requires individual framing. This framing consists of a guard time at the beginning and end of the slot, as well as a synchronizing preamble preceding the actual data. Therefore, if

multiple slots are assigned to the same mobile in one frame, some bandwidth is wasted on multiple slot boundaries within a *de facto* single transmission unit.

Some TDMA protocols, assume that individual bursts (talkspurts) in a CBR (voice) session are separate "sessions" of their own, which have to individually request bandwidth as they occur. With this approach, it may be possible to transmit non-CBR traffic during silent periods of a CBR session [5]. One problem with such protocols is that they admit the possibility of a collision when the voice source resumes its talkspurt, and the station may lose its reservation in such a case. An alternative approach is to keep the CBR slot reserved for the entire duration of the actual voice session. Of course, this approach is more wasteful because the bandwidth used to sustain a silent CBR session, although formally not needed, cannot be reused.

To be able to reuse the bandwidth temporarily relaxed by a silent CBR session, the base station must receive advance notifications about the status of the current burst. The concept of piggybacking [15, 16] seems to be an essential prerequisite for this capability, unless a separate signaling channel is used [9], which requires additional transmitters, receivers, and radio bandwidth. For example, if the only way for the base station to learn that the current burst has ended is to receive an empty CBR slot (like in PRMA or D-TDMA), the empty slot is irreparably wasted. On the other hand, if the stations piggyback their advance bandwidth requirements onto transmitted (full) slots, a CBR source can indicate that the burst is about to end early enough for its next slot to be reassigned to another mobile. This feature must be combined with a way of reverting a silent CBR session to its active state at the beginning of the next burst. Although the bandwidth scheduler at the base station can preempt less important sessions to make room for the new burst, it must be able to learn that the burst is there, i.e., that the silent session has become active. Ideally, the mobile should keep a tiny portion of its original bandwidth (slot) while in the silent state—just enough to be able to "piggyback" the new status of its session. With the rigid organization of the frame (where bandwidth comes in coarse slots) this approach is not feasible. A compromise solution may be to give a resumed CBR session a head start when competing for access to the base [10, 16]. With this approach, the protocol may provide an acceptable bound on the amount of time needed by a CBR source to notify the base about the status change of its session.

One flexible wireless scheme proposed in the literature is the Magic WAND [19]. Our proposed protocol may appear somewhat similar to the Magic WAND, especially that we also use ATM cells as the (sample) granularity of bandwidth allocation. However, in contrast to

the elaborate framing and bandwidth scheduling schemes of the Magic WAND (aimed at providing a comprehensive wireless solution for ATM networking), our simple solution should be viewed as an attempt at isolating the most important features of differentiated QoS in wireless TDMA systems and providing a simple and generic framework for flexibility and efficiency, not necessarily coupled to ATM.

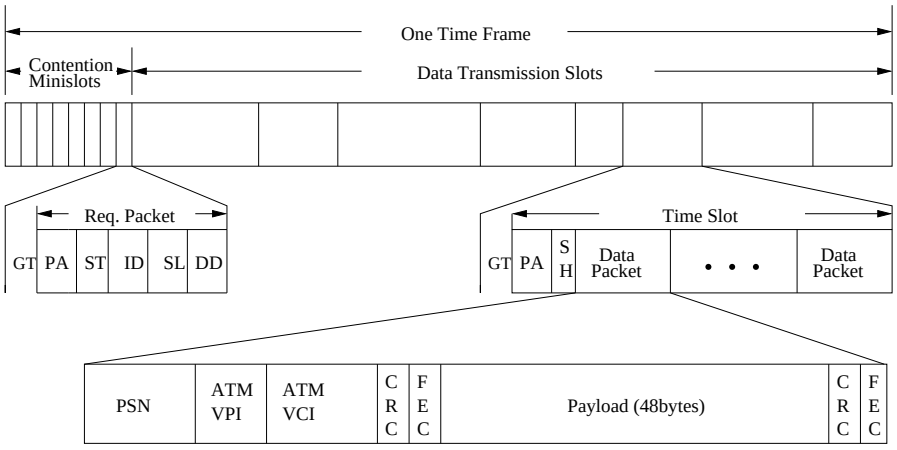
3. DS-TDMA/CP: protocol description

For our discussion, we assume that the downlink channel is separate from the uplink channel and that its implementation is uninteresting from the viewpoint of the access scheme needed for the uplink channel. In contrast to the protocols discussed in [10, 11, 15, 19], the uplink and downlink frames in our solution overlap in time.

Our proposed protocol is dubbed DS-TDMA/CP, for *Dynamically Slotted TDMA with Contention Permission*. Time on each of the two channels is divided into equal length frames. The frame length is set to coincide with the packet arrival rate of the CBR traffic. We assume that traffic in the network is naturally packetized with some granularity that is also used for bandwidth allocation. For the sake of discussion, we assume that this granularity is constrained by ATM cells, i.e., the network carries ATM traffic. However, this is not a requirement, and our solution can be naturally applied to non-ATM networks. Also, the bandwidth allocation grain can be arbitrarily fine (e.g., much finer than ATM cells) with little impact on the complexity of the overall scheme.

The uplink frame structure used by DS-TDMA/CP is shown in Figure 1. The frame is divided into two sections. The first (contention) section consists of a sequence of minislots used by the mobile stations to issue access requests. The second (transmission) section is dynamically partitioned into a number of variable-length transmission slots, according to the current bandwidth assignment to the mobile stations.

Consider the following traffic classes: CBR (bursty traffic with constant bit rate and tight deadlines), RT-VBR (variable bit rate traffic with deadlines that are less tight than those of the CBR class), NRT-VBR (VBR traffic with loose deadlines), ABR (higher priority traffic delivered according to a best effort policy), and UBR (low priority data traffic). CBR (the highest priority class) is ranked 0, and UBR (the lowest priority class) is ranked 4. An access request packet sent in a contention minislot includes the traffic class (i.e., expected service type), the mobile ID, the requested length of the transmission slot, and the deadline (*due date*), after which the request should be dropped if it cannot be granted. Note that a CBR source need not specify its



GT: Guard Time
PA: Modem Preamble
ST: Service Type (CBR/rt−VBR/nrt−VBR/ABR/UBR)
ID: User Identification Number
SL: Time Slot Length
DD: Due Date
SH: Slot Header
PSN: Packet Sequence Number
ATM VPI: ATM Virtual Path Identifier
ATM VCI: ATM Virtual Channel Identifier
CRC: Cyclic Redundancy Code
FEC: Forward Error Correction

Figure 1. Uplink frame structure in DS-TDMA/CP

bandwidth requirements because they are assumed to be the same for all CBR sessions. The base station stores the incoming requests and grants them according to the criteria explained below.

A single transmission slot accommodates a number of ATM cells (see Figure 1) and begins with guard time and a synchronizing preamble. The end of the preamble indicates the moment when the mobile may commence its transmission. The sequence of cells transmitted within the slot is preceded by a header in which the mobile piggybacks its further requirements for bandwidth. In particular, if the station has been transmitting CBR packets and its burst has run out, it will indicate in the slot header that its slot should be released in the next frame. Similarly, a station sending VBR traffic will indicate in the slot header the requested size of the next slot and the due date for this request.

The structure of the next uplink frame is announced by the base station on the downlink channel a moment before the frame is started. This announcement consists of the following information: the contention permission flags (CPF) indicating which traffic classes are allowed to compete for bandwidth, the number of minislots in the frame (S), and the list of scheduling messages consisting of pairs: $<$*station Id*, *slot length*$>$. A traffic class is allowed to compete for bandwidth only if its permission flag in the last frame announcement was set. The role

of *CPF* is to selectively restrain some traffic types when bandwidth becomes scarce.

The actual information sent in a frame announcement message is a bit disordered, e.g., the list of scheduling messages precedes the permission flags and the number of minislots. Based on that information, the mobile stations have to perform some simple calculations to determine the actual succession of their slots in the forthcoming frame (Section 4.3). This is needed to simplify the operation of the bandwidth scheduler at the base station, so that it can expedite the frame announcement on time—before the frame becomes due.

The boundary between the two sections of a frame, solely determined by the number of minislots S, varies according to the load in the network. When the load is heavy, the base station will issue a frame with a short contention section and a restrictive setting of *CPF*—to accept high-priority requests only. In an extreme case, e.g., if the frame is tightly occupied by CBR sessions that have all become active, the number of minislots may be zero.[1] On the other hand, when the load is light and there is little to transmit, the base will issue frames with long contention sections—to make the contention easier and quicker to resolve.

Suppose that a previously idle mobile station gets a packet to transmit and its traffic class is allowed to compete for bandwidth in the forthcoming frame. The station will randomly select one minislot in the contention section of that frame and transmit a reservation packet. This packet may make it to the base station or be destroyed, e.g., by a collision with another reservation packet sent by some other station. Other (piggybacked) reservation requests may arrive in the headers of slots transmitted within the frame.

When the base station has received the header of the last transmission slot, it will be ready to process new requests and schedule bandwidth for the next frame. The base station will try to accommodate the successfully received reservation requests, based on the available air-time, the traffic class priority, the requested slot length, and the specified deadlines. Then the station will broadcast the layout of the next frame to the mobile stations.

If the mobile's request has made it to the base station, the station will receive a scheduling message, even if its requirement cannot be met at present. In such a case, the slot length field of that scheduling message is set to zero. This will inform the mobile that its request

[1] It practically never happens in reality, because even a heavily loaded frame usually contains a small fragmented leftover that is turned into contention minislots (see the scheduling algorithm in Section 4.2). For example, in our experimental setup (Section 6.1), the minimum number of contention minislots was effectively 6.

has been noticed. The mobile will refrain from further contention for the amount of time corresponding to the specified time-out (due date) of the request. Similarly, if the base station cannot fulfill this request before the due date, it will drop the timed-out request from its queue. On the other hand, if the request packet has been lost (e.g., due to a collision), the mobile station will receive no scheduling message. Then it will know that the request should be retransmitted.

Note that all scheduling messages should be able to reach the mobile stations before the beginning of the next frame. The amount of time during which the base station has to make the scheduling decisions and transmit the packet announcing the layout of the upcoming frame is bounded by the length of the last slot in the current frame plus the inter-frame space. To maximize this time, the longest slot is always scheduled at the end of the frame.

By allocating the bandwidth for the next uplink frame at the end of the previous frame, the base station is able to account for as much air-time in the next frame as possible, which in turn results in a high flexibility and promptness in granting requests of the mobile stations. This is because those mobile stations that have been allotted space in the current frame piggyback their new bandwidth requests, including bandwidth relaxation, in slot headers.

Having received a scheduling message, a mobile station will start transmitting data packets in its allocated slot. If the traffic is non-CBR, the station will monitor its output buffer during transmission and piggyback slot adjustment requests in the header of its slot. If the buffer drains faster than the traffic is generated, the mobile will release some fraction of its slot. If the buffer grows faster than the data is being expedited, the station will request more bandwidth.

Once a CBR connection has been set up, the mobile will keep receiving a slot in every subsequent frame until the connection is explicitly torn down by the mobile. Instead of using the headers of its slots to convey bandwidth adjustment requests, a CBR source indicates there the status of its connection for the next frame: *active*, meaning that the burst (talkspurt) is still on and the next slot will be filled, *hung up*, meaning that the connection is terminated and the slot should be released, and *silent*, indicating that the connection should remain established but the next slot will be empty. In the last case, the base station is free to temporarily reduce the slot size in subsequent frames to the bare header. When the connection gets back to the *active* state, the mobile will use that header to indicate the state change to the base station. This way the mobile station will not have to contend for bandwidth when the silent period is over.

4. Detailed algorithms in DS-TDMA/CP

4.1. Contention resolution

The contention resolution part of DS-TDMA/CP can be implemented independently of the remaining elements of the protocol. We opt for a variant of slotted ALOHA inspired by [6, 16]. When a mobile station is ready to issue a request, it randomly selects a contention minislot and transmits the reservation packet. If the request makes it to the base, the mobile will receive a scheduling on the downlink channel. If this doesn't happen within the current frame, the station will assume that a collision has occurred and make another try (in the contention section of the next frame) with the probability

$$P_n = \min\left\{\frac{S_n}{S_c} \times \frac{P_c}{P_c + 1}, 1\right\}$$

where P_c is the retransmission probability used for the previous request (set to 1 for the first attempt), S_n is the total number of contention minislots available in the upcoming frame, and S_c is the number of contention minislots in the last frame. This backoff algorithm resembles the harmonic backoff described in [5, 16], and additionally accounts for the dynamic structure of the contention section.[2]

Needless to say, other collision resolution protocols, e.g., based on tree splitting [3, 16, 24], may be good alternatives to ALOHA. Although the performance of DS-TDMA/CP may somewhat vary depending on the collision resolution scheme, the protocol may still be compared with other similar solutions, as long as they all use similar methods of resolving contention.

4.2. Transmission scheduling

Each of the traffic classes listed in Section 3 has its specific QoS expectations; therefore, the access requests arriving at the base station are stored in different queues directly corresponding to the traffic classes. Requests in each queue are stored in the nondecreasing order of their deadlines. Requests with the same deadlines (e.g., UBR requests with infinite deadlines) are stored in the order of their arrivals. The scheduler tests the access requirements of the pending requests to see which of them can be accommodated and moved to the queue of scheduled requests. While doing this, the scheduler generates scheduling messages, one message per each request that has been granted.

[2] Our backoff function reduces to the harmonic backoff, if the length of the contention section remains fixed across frames.

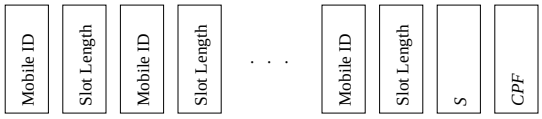


Figure 2. Structure of the frame announcement message

A non-CBR source that cannot finish its transmission in the current frame (which fact is signaled by a piggybacked request in the header of the currently transmitted slot) is put back into the corresponding access request queue. On the other hand, a CBR source, once it is admitted by the scheduler, will be receiving a guaranteed slot[3] in subsequent frames until the end of its session.

The amount of time available to the scheduler to complete its task and broadcast the layout of the new frame is limited by the size of the last slot in the current frame. To make a good use of this limited time, the scheduler generates the layout packet on-the-fly, while performing bandwidth allocation. Whenever a slot space is assigned to a mobile, the scheduler emits a scheduling message (Figure 2) identifying the mobile and specifying the size of the allocated slot. The allocation loop of the scheduler can be exited in one of two ways. First, it may happen that there is nothing more to do, i.e., all the available bandwidth has been allocated or all requests have been granted. Second, a timer may go off indicating that the scheduler must finish immediately because the frame is due. Intentionally, the second case should occur very seldom and possibly never, depending on the processing power available at the base station. We admit it because the protocol can operate sensibly in such circumstances. The worst that can happen is that some low-priority requests may not be fulfilled, even if enough bandwidth remains available to accommodate them.

As CBR requests have the highest priority, the scheduler processes them first, in the nondecreasing order of their due dates, according to the STE (shortest time to extinction) policy [20]. When all CBR requests have been processed, and there is still time and bandwidth left in the frame, the scheduler will process RT-VBR requests (second highest priority), and so on.

Having completed the allocation loop (and generated all scheduling messages), the scheduler emits two more items of information (Figure 2): the number of contention minislots in the upcoming frame (S) and the contention permission flags (CPF).

The protocol specifies a maximum on the number of contention minislots in a frame. Note that even if the network is completely idle,

[3] Possibly shrinking to the bare header during silent periods.

the frame cannot be entirely filled with minislots because the last few of them would be too close to the frame boundary to be usable by the scheduler. The actual number of contention minislots (which is never greater than the maximum) and the contents of *CPF* are determined by how the frame space has been partitioned among the multiple traffic classes and how much bandwidth (if any) remains unused. To understand the rationale behind this part of the scheduling algorithm, imagine two equally heavily loaded frames, one filled with UBR cells and the other carrying exclusively CBR traffic. Even though the two frames may look identical from the viewpoint of bandwidth utilization, it is clear that the numbers of minislots, as well as the settings of *CPF*, in them should be different. The second frame is filled with a high priority traffic, so it makes sense to restrain further contention with a restrictive setting of *CPF* and a reduced number of minislots. On the other hand, the low priority traffic in the first frame is preemptible by all other traffic classes. Those higher priority classes should be allowed to contend for bandwidth, for which they need more minislots and a permissive setting of *CPF*.

The scheduling algorithm

We assume the following notation for the constants and variables used by the algorithm:

Constants (protocol parameters):

- B — The total amount of bandwidth (frame space) available in an empty frame.
- B_u — The amount of bandwidth needed to sustain a silent CBR session. This bandwidth corresponds to the bare CBR slot header.
- B_s — The amount of bandwidth used up by one contention minislot.
- C_{max} — The maximum index of a traffic class, i.e., 4 according to Section 3. There is nothing special about this number, and it may change, if the configuration of traffic classes serviced by the network is different.
- S_{max} — The maximum number of minislots in a frame.
- B_v — This is an array indexed by traffic classes indicating the average amount of bandwidth needed by a session in the given class (see the explanation at the end of the algorithm). $B_v[0]$

is exact and equal to the fixed amount of bandwidth needed by an active CBR session.[4]

S_{incr} This is an array indexed by traffic classes used to determine the number of extra minislots to be included in the contention section of the next frame, as explained at the end of the algorithm. The entries corresponding to the first two traffic classes, i.e., $S_{incr}[0]$ and $S_{incr}[1]$, are not used.

Important variables:

S The calculated number of minislots in the upcoming frame.

CPF This is a bit array indexed by traffic classes and representing the calculated setting of the contention permission flags.

B_q The amount of bandwidth temporarily released by those admitted CBR sessions that are currently silent.

B_a The calculated amount of bandwidth still available within the frame, including the bandwidth temporarily released by the silent CBR sessions.

B_{asg} This is an array indexed by traffic classes and indicating how much bandwidth has been allocated to the given traffic class.

The algorithm

1. Set $S := 0$, $B_a := B$, $B_q := 0$, $B_{asg}[0, \ldots, C_{max}] := 0$, $CPF[0, \ldots, C_{max}] := 1$.

2. For all CBR sessions in progress, and then for all pending CBR requests (sorted in the non-decreasing order of their due dates), perform steps 3 through 6, with M_r indicating the mobile source.

3. If there are no more CBR requests to process, proceed to 7. If $B - B_{asg}[0] \geq B_v[0]$ (more CBR sessions can be accommodated), proceed to 4. Otherwise (this is only possible for a pending request), if the request arrived in the last frame, issue the scheduling message $< M_r, 0 >$. Continue at 3 for the next CBR request.

4. If the session is currently silent (this is only possible for a session in progress), set $B_q := B_q + (B_v[0] - B_u)$, $B_r := B_u$. Otherwise (including the case of a pending CBR request being admitted), set $B_r := B_v[0]$.

[4] Although the present algorithm assumes that every active CBR session uses the same fixed amount of bandwidth, this assumption can be easily relaxed.

5. Set $B_a := B_a - B_r$ and $B_{asg}[0] := B_{asg}[0] + B_v[0]$.

6. Transmit the scheduling message $< M_r, B_r >$ and continue at 3 for the next CBR session or request.

7. If $B - B_{asg}[0] < B_v[0]$, i.e., no more CBR sessions can be accommodated (this is only possible after all sessions in progress have been accounted for), set $CPF[0] = 0$. Otherwise, set $S_r = \lfloor (B - B_{asg}[0])/B_v[0] \rfloor$ (the number of extra minislots for the CBR sessions that could be accommodated in the frame), $B_a := B_a - S_r \times B_s$ (extract the bandwidth needed for the additional minislots), $S := S + S_r$.

8. Process the remaining traffic queues according to their priorities, and each queue in the non-decreasing order of the due dates. For each request, perform steps 9 through 13, with M_r indicating the the mobile source that issued the request, C_r indicating the traffic class, and B_r indicating the requested bandwidth.

9. If the internal timer went off, which means that there is no time for more scheduling before the frame is due, exit the allocation loop and continue at 14.

10. If there are more requests in the current class C_r, proceed to 11. Otherwise, if $B_a + B_{asg}[C_r] < B_v[C_r]$, set $CPF[C_r] = 0$. If $C_r = C_{max}$, proceed to 14. Otherwise, continue at 9 with $C_r := C_r + 1$.

11. If $C_r > 1$, calculate the number of additional contention minislots to be included in the frame: $S_r = \lfloor (B_{asg}[C_r] + B_r)/S_{incr}[C_r] \rfloor - \lfloor B_{asg}[C_r]/S_{incr}[C_r] \rfloor$, and the resulting total frame space requested: $B_d := B_r + S_r \times B_s$.

12. If $B_d > B_a$ (there is not enough bandwidth available), proceed to 13. Otherwise (the request can be granted), update $B_{asg}[C_r] := B_{asg}[C_r] + B_r$ (the amount of bandwidth allocated to the traffic class), $B_a := B_a - B_d$ (remaining frame space), $S := S + S_r$ (the number of minislots in the frame). Transmit the scheduling message $< M_r, B_r >$. Then continue at 9 for the next request.

13. There is no bandwidth available to grant the current request. If this request arrived in the last frame (i.e., it is a new request), then transmit the scheduling message $< M_r, 0 >$. Continue at 9 for the next request.

14. Exit from the allocation loop. Calculate the final number of minislots in the frame: $S := min\{S + (B_a/B_s), S_{max}\}$. Transmit $< S, CPF >$ and terminate.

The scheduler makes sure that all admitted CBR sessions can always be accommodated into the frame, even if they all become active simultaneously. Therefore, when a new CBR request is being scheduled, the available bandwidth B_a is decremented by B_q, i.e., the amount of bandwidth temporarily released by the silent CBR sessions. This extra bandwidth is included in B_a, and it can be reused by other (non-CBR) sources without problems, as they never make reservations for more than one frame.

After all CBR requests have been processed (step 7), the number of minislots in the new frame is incremented by the number of additional CBR requests that could be accommodated. If no more CBR sessions could be admitted, $CPF[0]$ is cleared to inhibit new CBR requests. Owing to the short due dates of CBR sessions and their relatively long duration, it makes little sense to admit and store new CBR requests if the sessions in progress have used up the entire bandwidth.

When allocating bandwidth to a non-CBR traffic class, the scheduler increases the number of minislots in the frame proportionally to the amount of bandwidth allocated to the class (step 11), to account for the fact that there exist higher priority classes that should be allowed to compete for that bandwidth. This does not happen when bandwidth is assigned to class number 1 (RT-VBR), because the extra minislots for the single class preceding it (CBR) are handled separately (step 7).

Except for that special case, any bandwidth assigned to a lower priority traffic class i is considered available by all classes $< i$. Therefore, such an allocation should result in additional contention opportunities (extra minislots) available to the higher priority classes (appropriately restricted by CPF). Note that the number of minislots must be calculated on-the-fly, as the amount of bandwidth available at any moment (B_a) depends on that number. $S_{incr}[i]$, for $2 \leq i \leq C_r$, specifies the amount of bandwidth assigned to class i that will enlarge the contention section by one additional minislot.

The optimum size of the contention section depends on the population of the contending stations and their specific bandwidth requirements. Therefore, although the issue can be studied in more depth, it is impossible to prescribe a single scaling formula that would be optimal in all circumstances. Besides, the absolute accuracy of this formula seems to be of secondary importance from the viewpoint of the overall quality of the discussed access scheme. With the simple heuristics used in our implementation of the protocol (Section 6.1), we assume that we know the average amount of bandwidth $B_v[i]$ allocated to a session in class i. Consider a situation in which some bandwidth B is allocated to an NRT-VBR session (class number 2). From the viewpoint of the RT-VBR class, this bandwidth is available; therefore, it makes sense

to increase the size of the contention section by $B/B_v[1]$ minislots—to accommodate that many new RT-VBR contenders. If the same amount of bandwidth B is allocated to a session in class $i > 2$, we should account for the fact that there are several classes more important than i, and each of them has its specific average bandwidth requirements. Thus, we use the following harmonic formula:

$$S_{incr}[i] = \frac{1}{\sum_{j=1}^{i-1} \frac{1}{B_v[j]}}\ ,\ 2 \le i \le C_{max}$$

which has the intuitively desirable property that $S_{incr}[2] = B_v[1]$ and $S_{incr}[i] < S_{incr}[i-1]$ for $2 < i \le C_{max}$. Note that, instead of being constants, the values $B_v[i]$, $1 \le i \le C_{max}$, can be updated on the fly, e.g., using an exponential averaging formula, depending on the measured actual amount of bandwidth allocated to sessions in a given class.

4.3. Processing at the mobile station

The information broadcast by the base station in the frame announcement packet must be preprocessed by the mobile stations. Consider a mobile station receiving an announcement packet. The sequence of scheduling messages $< M_r, B_r >$ arrives before the number of minislots, so the station must wait until the very end of the packet before it can know the exact positions of slots. The following algorithm lets the station learn the exact parameters of its slot:

1. Set $P := 0$ (the current position within the frame), $P_s := -1$ (the position of your slot), $B_{max} := 0$ (the maximum length of a slot seen so far).

2. For all subsequent scheduling messages $< M_r, B_r >$, perform steps 3 through 5, and proceed to 6 when done.

3. If M_r indicates this station, set $P_s := P$, $B_s := B_r$.

4. If $B_r > B_{max}$, set $B_{max} := B_r$, $P_{max} := P$.

5. Set $P := P + B_r$ and continue for the next scheduling message.

6. Read $< S, CPF >$, i.e., the number of minislots and the contention permission flags, and calculate $P_m = S \times B_s$, where B_s is the amount of frame space needed for one minislot (see section 4.2). P_m determines the length of the contention section.

7. If $P_s = -1$, terminate. This means that no bandwidth has been allocated to this station. If the request was previously accepted (see step 8), the station must continue waiting until the due date and reissue the request, if it isn't accepted by then. Otherwise, if the request was issued in the previous frame, it didn't make it to the base and must be reissued, according to the contention algorithm (Section 4.1).

8. If $B_s = 0$, terminate. This means that the request has made it to the base, but it hasn't been granted in this frame (see step 7).

9. The longest slot is moved to the end. If $P_s > P_{max}$ (our slot is located above the maximum length slot), set $P_s := P_s - B_{max}$; else if $P_s = P_{max}$ (our slot is the maximum length slot), set $P_s := P - B_{max}$.

10. Offset the slot pointer by the amount of space used by the minislots: $P_s := P_s + P_m$.

If all mobile stations execute the same algorithm, they will all come up with the same layout of the new frame, including the location of the longest slot, which is implicitly moved to the very end of the frame. If the frame contains more than one slot with the same maximum length, the first of them (according to the order of scheduling messages in the announcement packet) is moved to the end.

4.4. Scheduling UBR traffic

For a traffic class other than UBR, the *requested bandwidth* specified by the source in its request packet is treated by the scheduler as the exact amount to be allocated. Thus, it directly corresponds to B_r, as used in the scheduling algorithm discussed in Section 4.2. The interpretation of the requested bandwidth for UBR traffic is slightly different. The scheduler assumes that a UBR source can sensibly use any bandwidth whatsoever, and the specification arriving in the request packet is only viewed as the maximum.

Several fair scheduling strategies for UBR traffic are possible. With the simplest strategy (requiring the least amount of work from the scheduler), the outstanding UBR requests are processed in a round-robin fashion, and each of them receives as much bandwidth as available within the currently scheduled frame, up to the requested maximum. This approach, dubbed the *least effort policy* (LEP), minimizes the fragmentation of UBR packets into slots (individual UBR slots are as large as possible), and thus maximizes the throughput (frame utilization), but incurs the longest delays. Depending on the true nature of

the UBR applications, and the number of active UBR mobiles, this delay may be acceptable or not. Generally, due dates for UBR requests can be set very late, because there are no inherent timing constraints for this type of service.

On the other end of the spectrum is the *maximum population policy* (MPP), which maximizes the number of mobiles that can be serviced within one frame. With this approach, UBR slots will tend to be shorter (incurring more bandwidth overhead), but the delays perceived by individual users will be shorter as well.

The entire spectrum can be described by a single policy parameterized by the *preferred granularity* of bandwidth allocation, which we shall denote G_{ubr}. The idea is to allocate the UBR bandwidth in chunks of G_{ubr} cells, up to the amount requested by the source. Any leftovers are first spread among the sources that have already received bandwidth (but still need more), and then assigned to the remaining stations with the granularity as close to G_{ubr} as possible. If $G_{ubr} = \infty$, the resulting policy is LEP, if $G_{ubr} = B_{min}$, where B_{min} corresponds to the amount of bandwidth needed to accommodate a single ATM cell, we get MPP.

5. Traffic models

To study the performance of DS-TDMA/CP, we consider three traffic classes: CBR, which receives a special treatment and therefore must be included in any study, VBR, as the highest-priority non-CBR traffic pattern, and UBR exemplifying data traffic with the lowest priority. The CBR traffic is assumed to be voice, the VBR traffic represents video, and the UBR traffic corresponds to file transfers, i.e., data, without stringent delay requirements.

Voice traffic is commonly described by an "On-Off" model. According to this model, a voice source can be either in the *talkspurt* state, in which it generates data at a constant rate, or in the *silent* state, in which no data is generated at all. Our variant of this model closely follows the one considered in [12, 22]. The time spent in each state is exponentially distributed with two means: α^c for the "Off" state, and β^c for the "On" state. Following [18], we assume that the arrival process of new calls is Poisson. When the request of the mobile station is granted, a CBR session is set up between the mobile and the base. The mobile will be generating talkspurts interleaved with silence periods for as long as the session is not torn down.

For the VBR video traffic, we adopt the DAR(1) (Discrete Autoregressive) model described by three parameters: the mean, the variance,

and the first-order autocorrelation coefficient. The DAR(1) model was found in [13, 14] to characterize the variable bit rate of generic video sessions with a satisfying accuracy, and since then it has been used in many simulation studies, e.g., [4, 7]. In our model, the bandwidth specification of a VBR packet is always exact and must be met by the scheduler if the packet is to be accommodated.

A UBR session represents a file transfer and, once the request makes it to the base station, its due date is infinite. The transmission rate is flexible: the source can use whatever bandwidth is left over within the frame after the two higher priority traffic classes have been serviced. For as long as there remain data to be sent, the source will piggyback new bandwidth requests onto the transmitted packets. The base is free to assign to the mobile any bandwidth available within the frame.

6. Performance

The performance of DS-TDMA/CP has been investigated by simulation and compared to the performance of D-TDMA and DQRUMA—two generic representatives of the TDMA family. Our plan was to analyse the proposed solution with respect to the following performance aspects:

QoS tradeoffs. DS-TDMA/CP purports to cater to traffic classes with different priorities and QoS requirements. We would like to see how those multiple traffic classes share the network bandwidth and whether their actual handling by the protocol in terms of relative performance matches our intentions.

Responsiveness and flexibility. The primary idea behind delaying the scheduling decisions until the last possible moment is to accommodate as many requests as can be sensibly accommodated within the maximum delay of one frame. We would expect our protocol to be responsive to intermittent changes in load patterns and flexible with bandwidth allocation.

Bandwidth utilization. Owing to the variable slot size, DS-TDMA/CP should offer better effective throughput (i.e., smaller overhead) than solutions based on fixed size slots.

6.1. Experimental setup

We consider a network consisting of a single base station and a variable number of mobiles possibly trying to talk to the base at the same time. The transmission rate of the channel is 1Mb/s. The propagation delay

Table I. Network parameters

frame length (B)	60,000
slot guard time (GT)	8
request packet length (B_s)	60
maximum number of minislots (S_{max})	856
preamble length (PA)	32
slot header length (SH)	2-18[5]
packet serial number (PSN)	8
ATM VPI *and* VCI	16+16
CRC + FEC	8+8
ATM *payload*	384

of the channel is ignored; however, its impact is captured by the guard time included in the length of every slot and minislot. For uniformity, we express all parameters in bits, assuming that 1 bit = 1μs.

With reference to Figure 1 and Section 4.2, the exact numerical parameters assumed in our experimental setup are listed in Table I. The 60 bit request packet length results from putting together the packet preamble (32 bits), service type (3 bits), user identification (12 bits), requested slot length (8 bits), and due date (5 bits). Note that the slot length is quantized into ATM frames. Similarly, the due date is represented as 32 discrete levels that, in real life, can be transformed into whatever deadlines are considered sensible and natural for the collection of traffic classes handled by the network.

The length of the slot header (SH) field, which is used for piggybacking bandwidth requests, depends on the traffic class to which the slot has been allocated. For a CBR session, SH merely indicates the session state (active, silent, hung up—see Section 3), which information can be accommodated into 2 bits. For VBR and UBR sessions, the piggybacked bandwidth request must include a slot size and due date, i.e., the SL and DD fields from the request packet. Thus, for these traffic classes, the length of the SH field is 12 bits.

The slot length for an active CBR session (a talkspurt packet) is set to 2154 μs. The bit rate of a CBR session is assumed to 32 kb/s, with the mean durations of the talkspurt and silence periods settable as simulation parameters. Ignoring the overhead, a single CBR session requires 32/1000 of the channel time during its talkspurt phase, which translates into the same fraction of every frame. The CBR slot length includes that fraction as well as various overheads needed to turn the talkspurt frame into a transmittable unit.

The variance and autocorrelation parameters for the VBR traffic model are set to 5536 and 0.98, respectively [13]. The due date for all VBR packets is 150 ms. The mean source rate of a VBR session is assumed to be 128 kbps/s. The mean duration of a single VBR call is three minutes.

A UBR session represents a file transfer and, once the request makes it to the base station, its due date is infinite. However, a UBR request may expire while the source is waiting for access to the base station (i.e., permission to contend and a scheduling message). The expiration time, after which a UBR request is dropped at the source, is set at 30 seconds. The mean length of a file to be transferred is 104 KB. Note that because CBR sessions occasionally become silent, and VBR (as well as CBR) sessions have more rigid requirements regarding the slot size than UBR sessions, there usually is some bandwidth left for UBR traffic, even if the network is heavily loaded by CBR/VBR traffic. The G_{ubr} (preferred granularity) parameter for scheduling UBR traffic (Section 4.4) is set at three ATM cells.

6.2. QoS TRADEOFFS

The single most illustrative performance graph visualizing the properties of DS-TDMA/CP at a glance is Figure 3. The figure shows the fraction of the effective bandwidth used by each of the three traffic classes under increasing load conditions. The *load factor* of 0.8 indicates the average fraction of all mobiles (their total population is marked on the x-axis) involved in traffic sessions of the indicated type.

At the beginning, when the number of mobiles is small, there is enough bandwidth to accommodate all sessions without preemption. Then, the relative positions of the curves indicate the individual contributions of the three traffic types to the total offered load.[6]

When the system load reaches a certain threshold (15 mobiles, about 88% capacity), the service received by UBR requests begins to decline—to make room for CBR and VBR traffic. Note, however, that the system does not provide its maximum capacity at this point. Because all VBR and CBR sessions are satisfied, some bandwidth is set aside for additional contention minislots (Section 4.2). This mode of operation is sustained for as long as all VBR requests are still satisfied. Having a higher priority than UBR requests, they are first to reuse the silent slots left over by the CBR sessions in progress. The system reaches its maximum capacity around 25 mobiles, when some VBR sessions begin to be rejected.

[6] Note that as the traffic generation processes are not memory-less, it is impossible to maintain specific fixed proportions of offered load.

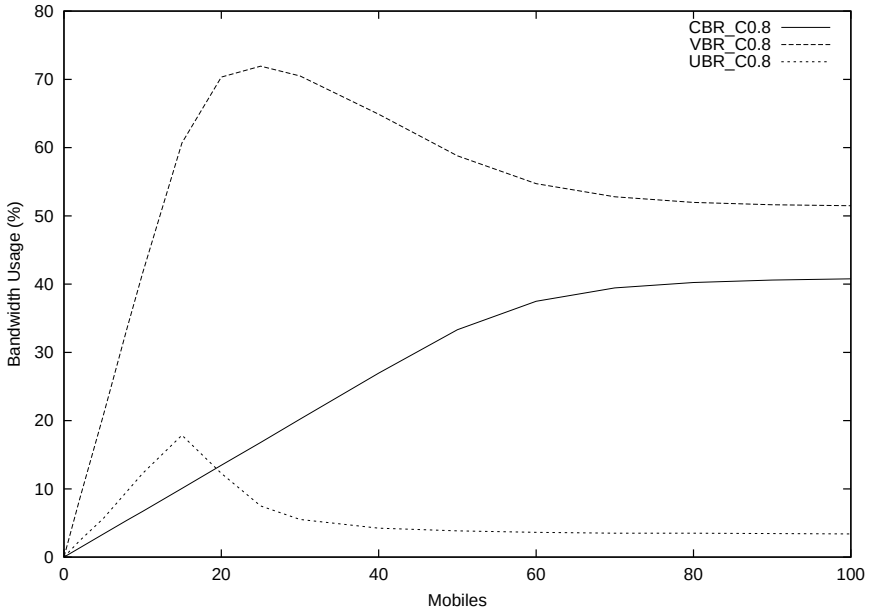


Figure 3. Bandwidth utilization in DS-TDMA/CP

The amount of CBR traffic accommodated by the bandwidth scheduler increases steadily with the load until its specific saturation threshold. The other two traffic types, i.e., VBR and UBR, yield to CBR, with UBR additionally yielding to VBR, although neither of them dies out. In fact, VBR stabilizes at a level that is considerably higher than the saturation threshold of CBR. This is because, as we explained in Section 4.2, CBR traffic is incapable of using more than a certain fraction of the entire useful bandwidth, roughly equal to $l_t/(l_t + l_s)$, where l_t and l_s are the average lengths of the talkspurt and silence periods. Similarly, UBR traffic is not completely eliminated by VBR. This is because VBR packets have much stricter bandwidth requirements than UBR packets, which can use practically any leftovers.

Although D-TDMA exhibits a bandwidth usage pattern similar to DS-TDMA/CP (see Figure 4), the CBR throughput curve has a steeper slope than in Figure 3. This is because the frame structure in D-TDMA is rigid and constrained by the channel rate and the source rate. Moreover, the CBR sessions in progress never make their silent periods available to other traffic classes. Although the maximum used CBR bandwidth reaches about the same level as in DS-TDMA/CP,[7]

[7] This is hardly surprising, as the frame structure in D-TDMA is tailored for CBR traffic.

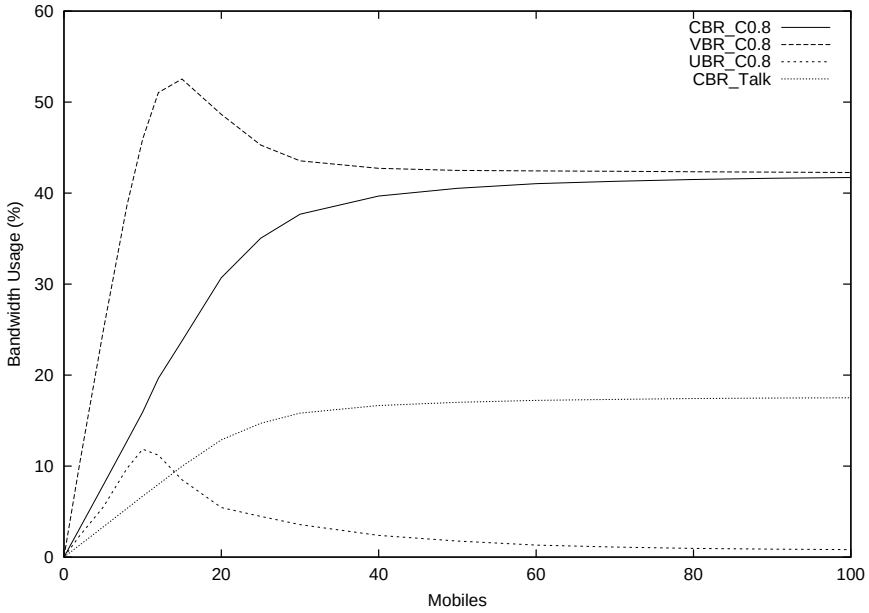


Figure 4. Bandwidth utilization in D-TDMA

the actually "used" portion of this bandwidth is about 18%, as shown by the dotted CBR curve that excludes the non-reusable silent periods. For this reason, the bandwidth assigned to VBR traffic (and the total useful bandwidth of the network) ends up at a considerably lower level than in DS-TDMA/CP.

In DQRUMA (Figure 5), the rigid bandwidth allocation scheme limits the performance in a completely different way. According to the protocol, access requests from all kinds of traffic are scheduled in a round-robin fashion. In our experiment, VBR is the traffic class with most requests. Consequently, within the light range of traffic conditions, when the system has enough bandwidth to accommodate all requests, VBR is getting more bandwidth than the other two classes. With the increasing load (from all classes), the round-robin policy tends to incur longer and longer scheduling delays. Some CBR and VBR requests (for which deadlines matter) time out and become dropped. The increased number of mobiles contending for their first transmission (and unable to piggyback their requests), push up the collision rate, which further reduces the amount of usable bandwidth. Since the VBR class has the highest number of requests, it is most adversely affected by this behavior, and the VBR bandwidth drops first. CBR sessions, needing fewer access requests to receive their share of bandwidth, are affected to a lesser extent. With the increasing number of CBR requests, the

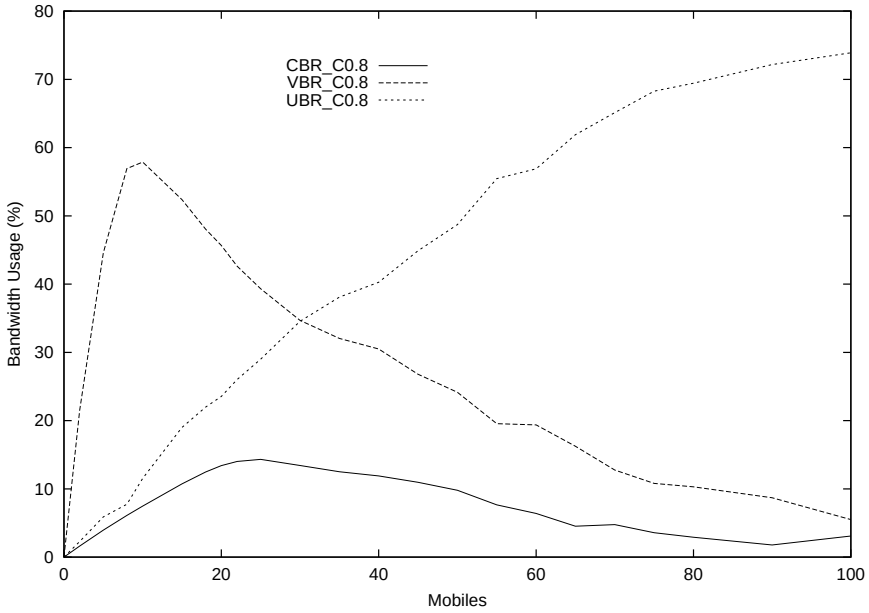


Figure 5. Bandwidth utilization in DQRUMA

bandwidth allocated to voice traffic continues to grow for a while after VBR has reached its maximum. Although the UBR traffic is also affected by the poor accessibility of bandwidth, those requests never time out. As long as an UBR request makes it to the base, it will get serviced, and the session will be able to sustain itself via the piggyback mechanism. This is why the bandwidth used by the UBR class keeps increasing to the very end of the investigated traffic range.

To demonstrate that the QoS received by CBR sessions in DS-TDMA/CP is not affected by the presence or absence of other traffic types, we carried out a series of experiments in which the offered CBR load remained steady, while the contribution of other traffic classes varied. The CBR load was set at a high level—to make the deviations of drop rate clearly visible, while the VBR and UBR load increased in proportion to the number of mobiles in the network. Figure 6 illustrates the stability of the CBR service in such circumstances for three different (two of them high) fixed levels of CBR load. The blocking rate is always a straight horizontal line,[8] which demonstrates that the CBR traffic has absolute priority in acquiring bandwidth.

Another illustration of the treatment received by different traffic classes in DS-TDMA/CP is given in Figure 7. It shows the bandwidth utilization in our network under traffic conditions similar to those

[8] The blocking rate for the load level of 0.2 is consistently zero.

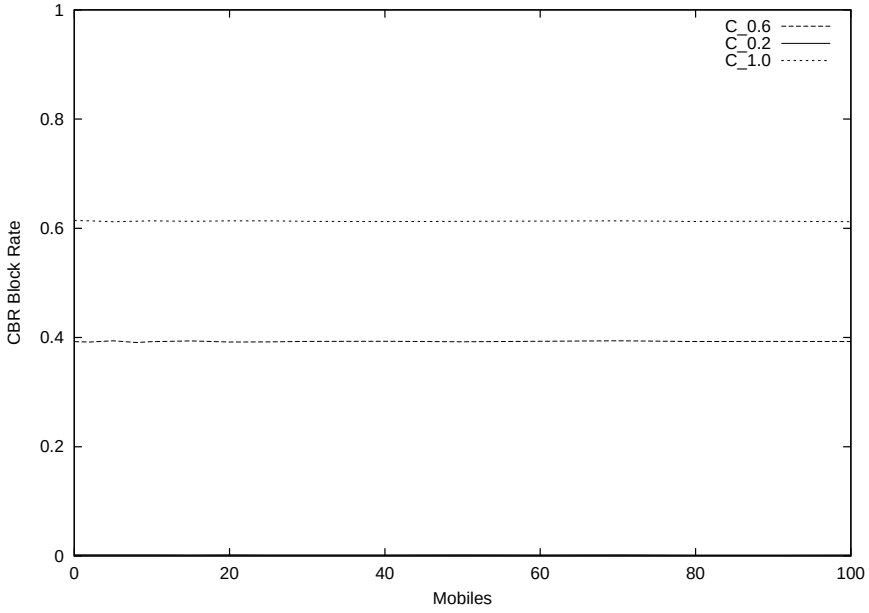


Figure 6. Blocking rate in DS-TDMA/CP for CBR traffic under constant CBR load

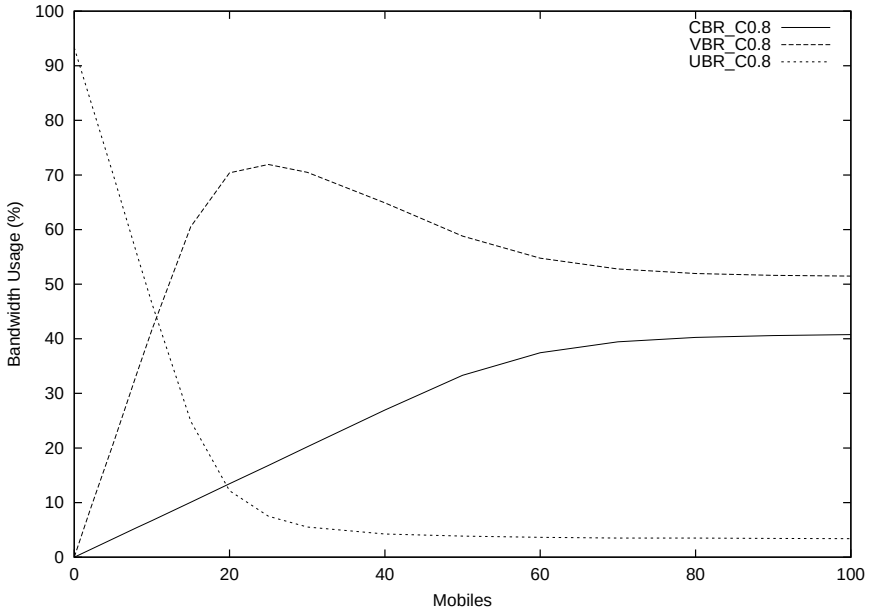


Figure 7. Bandwidth utilization in DS-TDMA/CP under constant UBR load

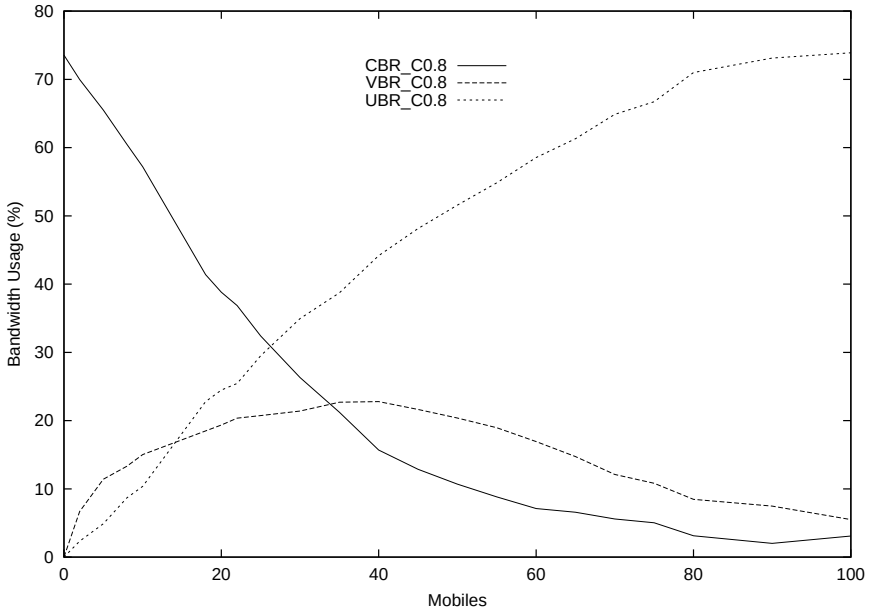


Figure 8. Bandwidth utilization in DQRUMA under constant CBR load

illustrated in Figure 3, except that the UBR load is kept constant at the highest level. The curves for VBR and CBR are completely indistinguishable from those in Figure 3.

Ignoring the numerical differences in achievable throughput, the behavior exhibited by D-TDMA under such conditions is very similar to DS-TDMA/CP, which is more than can be said about DQRUMA. Figure 8 was obtained under fixed heavy CBR load, with the remaining traffic classes behaving as in the experiment illustrated in Figure 5. Owing to the round-robin bandwidth allocation policy, the performance of DQRUMA for high-priority traffic is severely impaired by the presence of low-priority load. With the increasing UBR and VBR loads, the CBR traffic receives consistently smaller and smaller share of the network bandwidth. This happens because CBR sessions, having short deadlines, also have hard time getting through the contention stage. Consequently, they tend to time out and be dropped more often than other requests.

In fact, somewhat contradictory to its purpose, DQRUMA seems to favor UBR traffic, or, for that matter, any traffic class with long or infinite deadlines. Under all conditions, the amount of bandwidth allocated to UBR sessions tends to grow until it takes most of the available bandwidth. Any request whose deadline is sufficiently long is eventually

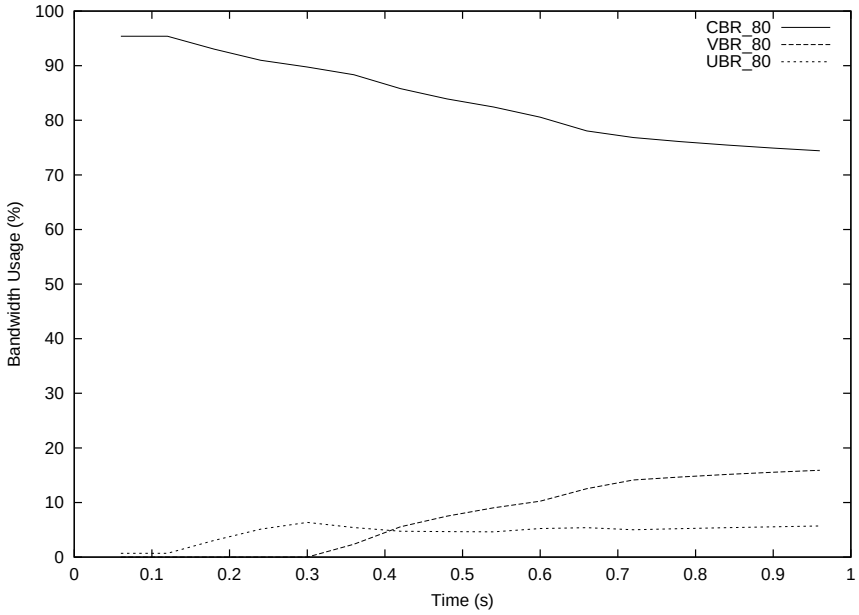


Figure 9. Burst response of DS-TDMA/CP, short talkspurt, heavy load.

able to make it to the base, and once that happens, the session will sustain itself through the piggyback mechanism of DQRUMA.

6.3. Burst responsiveness

Even though in the long term (at equilibrium) D-TDMA treats prioritized traffic classes in a way similar to DS-TDMA/CP, both D-TDMA and DQRUMA show a significantly slower responsiveness to rapid fluctuations in the offered load. This is illustrated in Figures 9–11. Owing to the fact that DS-TDMA/CP reuses silent periods in the CBR traffic, its high responsiveness to bursts is somewhat disguised in Figure 9 and needs a few words of clarification.

All three figures illustrate a sudden transition of the network from a completely idle state to being heavily loaded with traffic of all three classes (all sessions start up at the same time). The average duration of the talkspurt phase for a CBR session is equal to 1 second. For DS-TDMA/CP, the bandwidth used by the CBR class appears to drop for a while, which reflects the fact that some of the sessions switch to the silent phase (a CBR session always starts in a talkspurt). Note the initial flat portion of the CBR curve in Figure 9, which corresponds to the period during which all CBR sessions are still active. Within that period, the network behavior is stable from the very beginning, which is not the case with the remaining two protocols. Both D-TDMA and

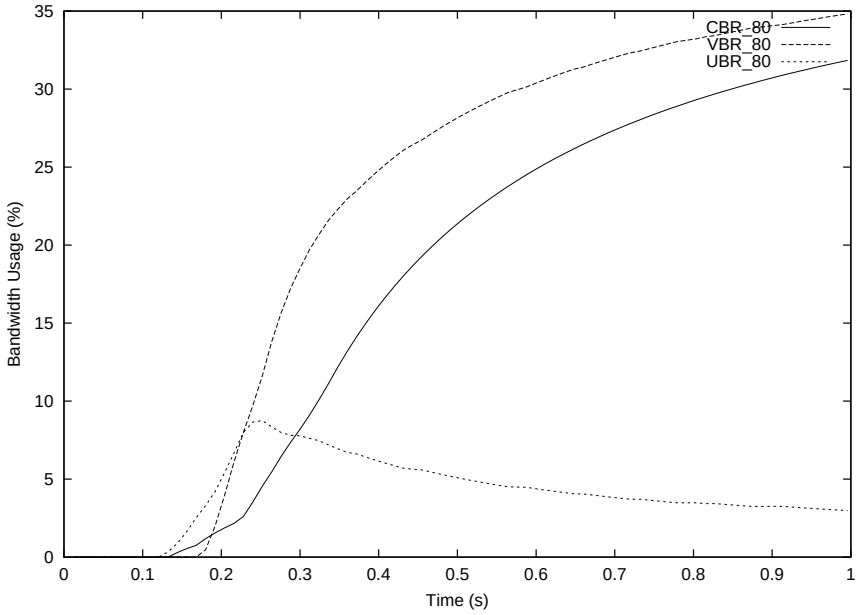


Figure 10. Burst response of D-TDMA, heavy load.

DQRUMA exhibit a considerably unstable behavior during an initial period of the burst.

The layout of the contention section of a frame in D-TDMA is fixed. Consequently, a considerable amount of bandwidth is wasted at the initial stage of burst resolution on coping with the contention among a huge number of requests arriving all the same time. DQRUMA suffers from a similar problem. Although, like DS-TDMA/CP, it tries to be flexible with the number of contention minislots, owing to the significantly shorter frame in DQRUMA, this number turns out to be inadequate for a quick resolution of the burst. Moreover, according to what we said above, once the network recovers from the burst, UBR takes most of the bandwidth, and CBR ends up with the least share. In contrast, when a large number of requests show up in an idle network driven by DS-TDMA/CP, the contention opportunities are much better than in DQRUMA and D-TDMA, and the burst is resolved incomparably faster.

6.4. Overhead

In comparing the bandwidth overhead incurred by the three protocols, we consider three mixes of traffic and three different bit rates for CBR sessions. Note that parameters like frame and slot size (for the fixed

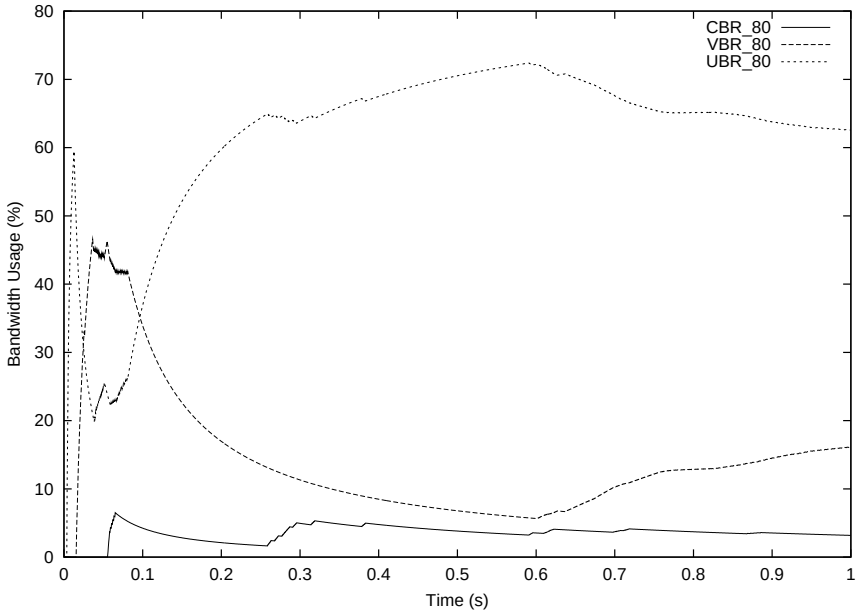


Figure 11. Burst response of DQRUMA, heavy load.

Table II. Bandwidth overhead for 1Mb/s channel

Protocol	CBR rate = 16kbps			CBR rate = 32kbps			CBR rate = 64kbps		
	M_C	M_V	M_U	M_C	M_V	M_U	M_C	M_V	M_U
DS-TDMA/CP	14.77	6.51	5.28	9.96	7.11	5.14	10.74	6.88	4.86
D-TDMA	15.18	12.22	12.22	15.21	15.21	15.20	17.68	16.26	15.20
D-TDMA*	50.53	18.36	18.88	39.49	24.58	24.58	41.90	31.90	30.84
DQRUMA	17.36	17.40	17.35	17.38	17.41	17.35	17.35	17.40	17.35

slot size protocols) are constrained by the channel rate, CBR rate and the CBR slot size.

Each of the three traffic mixes strongly favors one traffic type. With the mix denoted M_C, 90% of all traffic is contributed by CBR sessions, while the remaining traffic types (i.e., VBR and UBR) contribute 5% each. Similarly, mixes M_V and M_U favor VBR and UBR traffic types—in the same proportions. Reasonably accurate approximations of the overhead for other mixes can be obtained from the presented numbers by interpolation.

Table III. Bandwidth overhead for different channel rates

Protocol	rate = 0.5Mbps			rate = 1Mbps			rate = 1.5Mbps		
	M_C	M_V	M_U	M_C	M_V	M_U	M_C	M_V	M_U
DS-TDMA/CP	8.45	12.68	8.62	9.96	7.11	5.14	17.19	4.14	3.90
D-TDMA	15.20	15.33	15.20	15.21	15.21	15.20	15.23	15.20	15.20
D-TDMA*	39.53	30.93	31.03	39.49	24.58	24.58	39.48	21.45	21.45
DQRUMA	17.67	17.59	17.35	17.38	17.41	17.35	17.46	17.37	17.35

Table II lists the bandwidth overhead measured for the three investigated protocols under three traffic mixes and three CBR rates. The row labeled D-TDMA* includes the overhead caused by the silent periods in CBR traffic, which are unusable in D-TDMA. The overhead is calculated as the percentage of bandwidth of the uplink channel not used to transmit any data bits. Thus, it covers all guard intervals, gaps, headers and trailers. Table III lists the overhead measured for three different channel rates, assuming that the CBR session rate is 32 kbps.

Notably, for the traffic mix with predominant CBR component, the overhead incurred by DS-TDMA/CP is not much lower than for the other protocols,[9] although it tends to decrease with the increasing rate of CBR. The situation changes quite drastically, when the network load becomes dominated by non-CBR sessions. Note that for higher channel rates, the overhead of DS-TDMA/CP additionally tends to decrease, which demonstrates that the protocol well scales to the increasing transmission rate of the mobile network.

7. Conclusions

The TDMA protocol introduced in this paper offers differentiated quality of service to multiple traffic classes demanded by contemporary mobile applications. Owing to the variable and flexible slot size, our protocol incurs a lower bandwidth overhead than other solutions based on fixed size slots, especially when CBR traffic is not the dominant load in the network. By deferring scheduling decisions until the last possible moment and greatly improving contention opportunities under light load, the proposed access scheme is highly responsive to rapid changes

[9] Which have been designed with the CBR traffic in mind.

in the traffic pattern. This feature also makes it possible to identify and reuse silent periods in voice sessions to accommodate other traffic, without forcing the voice sessions to contend for bandwidth again at the beginning of a new talkspurt.

Our proposed solution poses only moderately higher demands on the complexity of the receivers than traditional schemes based on fixed-size slots. A receiver must store the allocated starting points of the slots expected to arrive in the next frame and use them to drive its timers. In contrast to a scheme based on fixed-size slots, those starting points cannot be "hardwired" into the timers which must be programmable. However, no extra bandwidth is going to be wasted on processing variable-length slots, and, of course, considerable bandwidth is going to be saved by eliminating unnecessary slot boundaries.

One problem with our approach to bandwidth scheduling is the relatively high demand on processing power at the base station. Although this demand appears to be well within the reach of contemporary hardware, its exact magnitude and dependence on the parameters of the network should be investigated and quantified. The implementation of the scheduling algorithm at the base station, including the design of data structures used for storing the request queues, deserves some studies as well. These issues will be addressed in further research.

References

1. Amitay, N.: 1993, 'Distributed Switching and Control with Fast Resource Assignment/Handoff for Personal Communication Systems'. *IEEE Journal on Selected Areas in Communications* **11**, 842–849.
2. Bianchi, G., F. Borgonovo, L. Fratta, L. Musumeci, and M. Zorzi: 1997, 'C-PRMA: a Centralized Packet Reservation Multiple Access for Local Wireless Communications'. *IEEE Transactions on Vehicular Technology* **46**, 422–436.
3. Capetanakis, J.: 1979, 'Tree Algorithms for Packet Broadcast Channels'. *IEEE Transactions on Information Theory* **25**, 505–515.
4. Chen, J., K. M. Sivalingam, and R. Acharya: 1998, 'Comparative analysis of wireless ATM channel access protocols supporting multimedia traffic'. *Mobile Networks and Applications* **3**(3), 293–306.
5. Choi, S. and K. G. Shin: 1998, 'A cellular wireless local area network with QoS guarantees for heterogeneous traffic'. *Mobile Networks and Applications* **3**(1), 89–100.
6. Choi, S. and K. G. Shin: 1999, 'An uplink CDMA system architecture with diverse QoS guarantees for heterogeneous traffic'. *IEEE/ACM Transactions on Networking* **7**(1), 616–628.
7. Dyson, D. A. and Z. J. Haas: 1997, 'A Dynamic Packet Reservation Multiple Access Scheme for Wireless ATM'. In: *Proceedings of IEEE MILCOM'97*. Monterey, CA.

8. Falk, G., J. Groff, W. Milliken, M. Nodine, S. Blumenthal, and W. Edmond: 1983, 'Integration of Voice and Data in the Wideband Packet Satellite Network'. *IEEE Journal on Selected Areas in Communications* **1**(6).
9. Fantacci, R. and S. Nannicini: 2000, 'Performance evaluation of a reservation TDMA protocol for voice/data transmission in microcellular systems'. *IEEE Journal on Selected Areas in Communications* **18**(11), 2404–2416.
10. Frigon, J., V. Leung, and H. Chan Bun Chan: 2001, 'Dynamic reservation TDMA protocol for wireless ATM networks'. *IEEE Journal on Selected Areas in Communications* **19**(2), 370–383.
11. Ganesh Babu, T., T. Le-Ngoc, and J. Hayes: 2001, 'Performance of a priority-based dynamic capacity allocation scheme for wireless ATM systems'. *IEEE Journal on Selected Areas in Communications* **19**(2), 355–369.
12. Goodman *et al*, D. J.: 1989, 'Packet Reservation Multiple Access for Local Wireless Communications'. *IEEE Transactions on Communications* **37**(8), 885–890.
13. Heyman, D., T. Lakshman, A. Tabatabai, and H. Heeke: 1994, 'Modeling teleconference traffic from VBR video coders'. In: *IEEE International Conference on Communications*, Vol. 3.
14. Heyman, D., A. Tabatabei, and T. Lakshman: 1992, 'Statistical analysis and simulation study of video teleconference traffic in ATM networks'. *IEEE Transactions on Circuits and Systems for Video Technology* **2**, 49–59.
15. Kang, C., C. Ahn, K. Jang, and W. Kang: 2000, 'Contention-free distributed dynamic reservation MAC protocol with deterministic scheduling (C-FD/sup3/R MAC) for wireless ATM networks'. *IEEE Journal on Selected Areas in Communications* **18**(9), 1623–1635.
16. Karol, M. J., Z. Liu, and K. Y. Eng: 1995, 'Distributed-queueing request update multiple access (DQRUMA) for wireless packet (ATM) networks'. In: *Proceedings of IEEE International Communications Conference*. Seattle, WA.
17. Kwok, Y. and V. Lau: 2001, 'A quantitative comparison of multiple access control protocols for wireless ATM'. *IEEE Transactions on Vehicular Technology* **50**(3), 796–815.
18. Markoulidakis, J. G., G. L. Lyberopoulos, and M. E. Anagnostou: 1998, 'Traffic model for third generation cellular mobile telecommunication systems'. *Wireless Networks* **4**(5), 389–400.
19. Mikkonen, J., J. Aldis, G. Awater, A. Lunn, and D. Hutchinson: 1998, 'The Magic WAND – functional overview'. *IEEE Journal on Selected Areas in Communications* **16**(6), 953–972.
20. Panwar, S. S., D. Towsley, and J. K. Wolf: 1988, 'Optimal scheduling policies for a class of queues with customer deadlines to the beginning of service'. *Journal of the ACM* **35**, 832–844.
21. Qiu, X. and V. O. K. Li: 1996, 'Dynamic Reservation Multiple Access (DRMA): A new multiple access protocol for Personal Communication Systems (PCS)'. *Wireless Networks* **2**(2).
22. Qiu, X., V. O. K. Li, and J. Ju: 1996, 'A multiple access scheme for multimedia traffic in wireless ATM'. *Mobile Networks and Applications* **1**(3), 259–272.
23. Wilson, N. D., R. Ganesh, K. Joseph, and D. Raychaudhuri: 1993, 'Packet CDMA versus dynamic TDMA for multiple access in an integrated voice/data PCN'. *IEEE Journal on Selected Areas in Communications* **11**, 870–884.
24. Yuang, M. and P. Tien: 2000, 'Multiple access control with intelligent bandwidth allocation for wireless ATM networks'. *IEEE Journal on Selected Areas in Communications* **18**(9), 1658–1669.